

CHAT OF DEATH - USER MANUAL

Anonymous P2P Encrypted Communication Application

Version: 1.0

Publisher: Death's Head Software

Last Updated: November 2025

TABLE OF CONTENTS

1. [Introduction](#)
 2. [Security Overview](#)
 3. [Prerequisites](#)
 4. [Initial Setup](#)
 5. [Establishing a Secure Connection](#)
 6. [Fingerprint Verification \(CRITICAL\)](#)
 7. [Using the Application](#)
 8. [Troubleshooting](#)
 9. [Security Best Practices](#)
 10. [Technical Specifications](#)
-

INTRODUCTION

Chat of Death is an anonymous, peer-to-peer encrypted communication application designed for maximum privacy and security. It uses:

- **Tor Hidden Services** for anonymity (no IP address exposure)
 - **End-to-End Encryption (E2EE)** using ECDH key exchange + AES-256-GCM
 - **Fingerprint Verification** to prevent Man-in-the-Middle attacks
 - **Encrypted Media Streaming** for voice and video calls
 - **Encrypted File Transfer** for secure document sharing
-

SECURITY OVERVIEW

What Makes Chat of Death Secure?

1. **Anonymous Routing:** All connections route through the Tor network, hiding your IP address and physical location.
 2. **End-to-End Encryption:** Messages are encrypted on your device and can only be decrypted by your intended recipient. Even if intercepted, they cannot be read.
 3. **Perfect Forward Secrecy:** Each session uses a unique encryption key. Past communications cannot be decrypted even if future keys are compromised.
 4. **Fingerprint Verification:** Ensures you're communicating with the intended person and not an attacker intercepting your connection.
-

PREREQUISITES

Required Software

1. Tor Browser or Tor Service

- Download from: <https://www.torproject.org> (<https://www.torproject.org>).
- Must be running before launching Chat of Death
- Default ports used:
 - SOCKS Proxy: 9150 (Tor Browser) or 9050 (Tor Service)
 - Control Port: 9151 (Tor Browser) or 9051 (Tor Service)

2. Python 3.8+ (if running from source)

3. Dependencies (if running from source):

```
pip install PyQt6 cryptography PySocks stem
```

Optional Dependencies

For full media functionality:

```
pip install opencv-python sounddevice
```

INITIAL SETUP

First Launch

1. Start Tor Browser/Service

- Open Tor Browser and wait for it to connect
- Keep it running in the background

2. Launch Chat of Death

- The splash screen will show initialization steps
- Wait for "System ready! Awaiting peer connection..."

3. Check Tor Connection

- Go to: **Help > Check Tor SOCKS Connection**
- Should show: "☐ Tor SOCKS Proxy connection successful"

4. Note Your Onion Address

- Located at the top of the **Rendezvous Key Exchange** tab
 - Format: `abcd1234...xyz.onion:8000`
 - This is YOUR address that peers use to connect to you
-

ESTABLISHING A SECURE CONNECTION

Step-by-Step Connection Process

PERSON A (Receiver/Server)

1. Share Your Onion Address

- Copy your onion address from the top of the Rendezvous tab
- Click "Copy My Onion Address" button
- Send this to Person B through a separate secure channel (Signal, ProtonMail, etc.)

2. Wait for Connection

- Chat of Death automatically listens for incoming connections
- When Person B connects, you'll see a status update

3. Exchange Public Keys

- Copy YOUR public key from the "Your Ephemeral Public Key" box (left side)
- Send it to Person B through a separate channel
- Paste PERSON B's public key into the "Peer Public Key (Paste Here)" box (right side)

4. Perform Key Exchange

- Click "2. Perform E2EE Key Exchange"
- Wait for "☐ E2EE Key Established"
- Your fingerprint will appear in the "Your Local" field

PERSON B (Initiator/Client)

1. Get Person A's Onion Address

- Receive Person A's `.onion:8000` address through secure channel

2. Initiate Connection

- Paste Person A's onion address in the "Address" field
- Port should be `8000` (default)
- Click "1. Initiate Anonymous Connection"
- Wait for "☐ Anonymous TCP Connection Established"

3. Exchange Public Keys

- Copy YOUR public key from the "Your Ephemeral Public Key" box (left side)
- Send it to Person A through a separate channel
- Paste PERSON A's public key into the "Peer Public Key (Paste Here)" box (right side)

4. Perform Key Exchange

- Click "2. Perform E2EE Key Exchange"
- Wait for "☐ E2EE Key Established"
- Your fingerprint will appear in the "Your Local" field

FINGERPRINT VERIFICATION (CRITICAL)

What Are Fingerprints?

A **fingerprint** is a unique identifier derived from your encryption session key. It's like a digital signature that proves you're communicating with the intended person.

Format Example:

12AB34CD 56EF78GH 90IJ12KL 34MN56OP 78QR90ST 12UV34WX 56YZ78AB 90CD12EF

Why Fingerprint Verification Is CRITICAL

Without fingerprint verification, you are vulnerable to **Man-in-the-Middle (MITM) attacks**, where an attacker intercepts your connection and pretends to be your intended recipient.

Scenario:

- You think you're talking to Alice
- An attacker intercepts and relays messages
- The attacker can read EVERYTHING
- You have no way to know unless you verify fingerprints

How Fingerprint Verification Works

1. **After Key Exchange:** Both parties see two fingerprints:

- **Your Local:** Your own fingerprint
- **Peer's:** The fingerprint received from the other person
- **This may take a moment.

2. **If Fingerprints Match:**

- Click "3. Verify and Trust Peer"
- Button turns green: "✅ PEER VERIFIED"
- All future messages show **[VERIFIED]** tag
- Connection is secure and authenticated

3. **If Fingerprints DON'T Match:**

- ⚠ **DANGER! DO NOT CONTINUE!**
- Click "3. Verify and Trust Peer" to see the warning
- Button turns red: "❌ FAILED - DO NOT TRUST"
- **Immediately disconnect**
- Possible MITM attack in progress
- **DO NOT send sensitive information**

USING THE APPLICATION

Text Chat Tab

After successful verification:

1. Type your message in the bottom text box
2. Press Enter or click "Send"
3. Messages appear in the chat history:
 - **Green text:** Your messages
 - **Blue text:** Peer's messages
 - **[VERIFIED] tag:** Appears if peer is verified

Meeting Room (Video/Voice) Tab

Hardware Setup:

1. Select Camera:

- Dropdown shows detected cameras
- Select "No Camera" to disable video

2. Select Microphone:

- Dropdown shows detected microphones
- Select "No Microphone" to disable audio

3. Test Devices:

- Click "Test Camera" to verify camera works
- Click "Test Microphone" to verify mic works

During Call:

- **Mute Toggle:** Click to mute/unmute microphone
 - ☐ ACTIVE = Microphone on
 - ☐ MUTED = Microphone off
- **Video Toggle:** Click to stop/start video
 - ☐ LIVE = Camera on
 - ☐ STOPPED = Camera off
- **Record:** Records encrypted video (saved as `.codrec` file)

File Share (Encrypted) Tab

Sending Files:

1. Click "Choose File to Send"
2. Select a file from your computer
3. Click "Send File (E2EE)"
4. File is encrypted and transmitted

Receiving Files:

1. Incoming files appear in the transfer list
 2. Files are automatically decrypted
 3. Saved to your Downloads folder
 4. Progress bar shows transfer status
-

TROUBLESHOOTING

"Tor SOCKS Proxy connection refused"

Problem: Tor is not running or not fully initialized.

Solution:

1. Open Tor Browser
2. Wait for "Connected to the Tor network"
3. Keep Tor Browser open in the background
4. Restart Chat of Death

"Connection Error: Rendezvous failed"

Problem: Peer's hidden service is not reachable.

Solutions:

- Verify the onion address is correct (should end in `.onion`)
- Ensure the peer has Tor running
- Ensure the peer's Chat of Death is open and listening
- Check port number is `8000`
- Wait a few seconds and try again (Tor circuits can be slow)

"Error. Paste peer's Public Key first"

Problem: You clicked "2. Perform E2EE Key Exchange" without pasting the peer's key.

Solution:

1. Get peer's public key from them
2. Paste into "Peer Public Key (Paste Here)" box (right side)
3. Try again

Fingerprints Don't Match

Problem: Possible Man-in-the-Middle attack or connection error.

Solution:

1. **Disconnect immediately**
2. Close Chat of Death
3. Restart Tor Browser (to get new circuit)
4. Restart Chat of Death
5. Try connecting again
6. If still doesn't match, **DO NOT USE** - your connection may be compromised

No Video/Audio

Problem: Hardware not detected or selected.

Solution:

1. Install optional dependencies:

```
pip install opencv-python sounddevice
```

2. Select correct camera/microphone from dropdowns
3. Click "Test Camera" and "Test Microphone"
4. Grant permissions if prompted by OS

SECURITY BEST PRACTICES

DO:

- ☐ **Always verify fingerprints** before sharing sensitive information
- ☐ **Use a separate secure channel** for verification (phone, video, in-person)
 - ☐ **Keep Tor Browser updated** to latest version
- ☐ **Use strong, unique nicknames** that identify you to your peer
- ☐ **Disconnect and reconnect** if anything seems suspicious
- ☐ **Close the application** when not in use
- ☐ **Save your recording files** in encrypted storage

DON'T:

- ☐ **Never skip fingerprint verification** - this is the most critical security step

- ❑ **Never verify fingerprints through Chat of Death** - always use out-of-band verification
 - ❑ **Never share your onion address publicly** - only with intended recipients
 - ❑ **Never reuse the same session** for different people - disconnect and reconnect
 - ❑ **Never assume encryption is enough** - encryption without authentication is vulnerable
 - ❑ **Never trust the connection** if fingerprints don't match
 - ❑ **Never share sensitive info** before verification is complete
-

TECHNICAL SPECIFICATIONS

Cryptography

- **Key Exchange:** Elliptic Curve Diffie-Hellman (ECDH) with SECP384R1 curve
- **Session Key Derivation:** HKDF (HMAC-based Key Derivation Function) with SHA-256
- **Symmetric Encryption:** AES-256-GCM (Galois/Counter Mode)
- **Key Size:** 256 bits (32 bytes)
- **Nonce Size:** 96 bits (12 bytes, randomly generated per message)
- **Fingerprint:** SHA-256 hash of session key (displayed as hex)

Network

- **Anonymity Layer:** Tor Hidden Services (.onion addresses)
- **Protocol:** TCP over Tor SOCKS5 proxy
- **Default Port:** 8000
- **Tor SOCKS Port:** 9150 (Tor Browser) or 9050 (Tor Service)
- **Tor Control Port:** 9151 (Tor Browser) or 9051 (Tor Service)

Data Types

- **CHAT (Type 1):** Encrypted text messages
- **MEDIA (Type 2):** Encrypted audio/video streams
- **NICKNAME (Type 3):** Encrypted nickname exchange
- **FINGERPRINT (Type 4):** Encrypted fingerprint exchange

Packet Format

```
[4 bytes: Length][1 byte: Type][N bytes: Encrypted Payload]
```

FREQUENTLY ASKED QUESTIONS

Q: Why do I need to use Tor?

A: Tor hides your IP address and physical location. Without it, anyone intercepting your connection can see who you're talking to and where you're located.

Q: Can the government/ISP see my messages?

A: No. Messages are end-to-end encrypted. Even if intercepted:

- They see encrypted data (looks like random bytes)
- They see you're using Tor (not illegal in most countries)
- They CANNOT see message content, who you're talking to, or what you're sharing

Q: What if I lose my encryption key?

A: Each session generates a NEW key. If you disconnect and reconnect, you'll get a new key and new fingerprints. Previous messages cannot be decrypted without the old session key.

Q: Can Chat of Death be used for group chats?

A: Not currently. Chat of Death is designed for secure 1-on-1 communication. Group chat would require different cryptographic protocols.

Q: Is my onion address permanent?

A: No. Each time you restart Chat of Death, a new ephemeral hidden service is created with a new `.onion` address. This provides additional anonymity.

Q: What happens if verification fails?

A: If fingerprints don't match, you're either:

1. Victim of a Man-in-the-Middle attack (attacker intercepting)
2. Experiencing a rare connection error **Either way: DO NOT CONTINUE. Disconnect immediately.**

Q: How do I know Chat of Death isn't compromised?

A: Chat of Death is open source. You can:

1. Review the code yourself
2. Compile from source
3. Verify fingerprints still protect you even if the software is compromised

Q: Can my peer fake their fingerprint?

A: No. The fingerprint is mathematically derived from the encryption key. If an attacker changes anything, the fingerprint will be different. This is why out-of-band verification is critical.

EMERGENCY PROCEDURES

If You Suspect Compromise:

1. **Stop communicating immediately**
2. **Disconnect** from Chat of Death
3. **Close Tor Browser** (gets new identity/circuit)
4. **Restart your computer** (clears memory)
5. **Scan for malware**
6. **Contact your peer through alternate secure channel**
7. **Start fresh connection** with new verification

If Verification Fails Multiple Times:

1. **Assume your network is compromised**
 2. **Switch to different network** (different WiFi, VPN, etc.)
 3. **Contact peer out-of-band** to confirm they're trying to connect
 4. **Consider your devices may be infected**
-

GLOSSARY

Anonymous Network: Network that hides participant identities and locations (e.g., Tor)

E2EE (End-to-End Encryption): Encryption where only sender and recipient can decrypt messages

ECDH (Elliptic Curve Diffie-Hellman): Method for two parties to agree on a shared secret key over an insecure channel

Fingerprint: Unique identifier derived from encryption key, used to verify connection authenticity

Hidden Service: Tor service with a `.onion` address, providing server anonymity

Man-in-the-Middle (MITM): Attack where attacker intercepts communication between two parties

Onion Address: Tor hidden service address (e.g., `abc123...xyz.onion`)

Out-of-Band Verification: Verifying information through a separate, independent channel

Perfect Forward Secrecy: Property where compromise of long-term keys doesn't compromise past session keys

Session Key: Temporary encryption key used for a single communication session

Tor: The Onion Router - anonymous communication network

SUPPORT AND CONTACT

For security vulnerabilities: Please report responsibly to the developer

For general questions: Refer to this manual first

Source Code: Available for review and audit

LEGAL DISCLAIMER

Chat of Death is provided as-is for legitimate privacy purposes. Users are responsible for:

- Complying with local laws regarding encryption and anonymity tools
- Using the software ethically and legally
- Understanding that no software provides 100% security
- Following proper security procedures (especially fingerprint verification)

The developers are not responsible for misuse, improper configuration, or failure to follow security best practices.

VERSION HISTORY

v1.0 (November 2025)

- Initial release
 - ECDH + AES-256-GCM encryption
 - Tor hidden service integration
 - Fingerprint verification
 - Media streaming support
 - Encrypted file transfer
-

END OF MANUAL

For the latest version of this manual, check the application's Help menu or source repository.

Remember: Security is only as strong as your weakest link. Always verify fingerprints!